AD-A041 747 MICHIGAN UNIV ANN ARBOR DEPT OF ELECTRICAL AND COMPU--ETC F/G 12/1
VECTOR PROCESSING IN SIMULATION.(U)
APR 77 D A CALAHAN AF-AFOSR-2812-75
UNCLASSIFIED AFOSR-TR-77-0800 NL

1 OF 1
ADA
041747

END
DATE
FILMED
7-77

ADA041747

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 77-0800	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) VECTOR PROCESSING IN SIMULATION.		5. TYPE OF REPORT & PERIOD COVERED Interim report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) D. A. Calahan		8. CONTRACT OR GRANT NUMBER(s) AF-AFOSR ~~75~~-2812-75
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of Michigan Dept. of Elect. & Computer Engineering Ann Arbor, Michigan 48105		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F 2304/A3
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research Bolling AFB, Washington, D. C. 20332		12. REPORT DATE April 22, 1977
		13. NUMBER OF PAGES 5
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) 22 Apr 77 6p.		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Proceedings Conf. on Modeling and Simulation
University of Pittsburg
April 21-22, 1977

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Parallel Processing
Vector Processing

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The availability of large scientific computers specially designed to solve problems 100-1000 times faster than current conventional processors will shortly open new opportunities to simulation-oriented research. This paper presents the attributes of problems commonly solved on such machines, presents simplified mathematical models and corresponding methods of evaluating their performance, and gives results of benchmark studies.

407 400

AFOSR-TR- 77-0800

Approved for public release;
distribution unlimited.

VECTOR PROCESSING IN SIMULATION

D. A. Calahan
Professor
Electrical and Computer Engineering
The University of Michigan
Ann Arbor, Michigan 48105

ABSTRACT

The availability of large scientific computers specially designed to solve problems 100-1000 times faster than current conventional processors will shortly open new opportunities to simulation-oriented research. This paper presents the attributes of problems commonly solved on such machines, presents simplified mathematical models and corresponding methods of evaluating their performance, and gives results of benchmark studies.

INTRODUCTION

Reading the titles of papers in this conference, one sees a concentration on the mathematics of simulation and its application to a variety of economic, social, environmental, and physical systems. Only a few sessions have to do with the tools of most simulation studies, i.e., the analog, digital, and hybrid computer.

This preoccupation is understandable in part because present machines have significant computational power and user convenience, allowing discipline-oriented users to be little concerned about design characteristics of the computer.

Recent advances in technology and computer design promise to significantly enlarge the size of simulation problems solveable in reasonable computation times. The new processors - usually termed vector or array processors - exploit regularity of a problem structure to achieve significantly faster computation speeds.

To particularize, recent benchmarks have shown that existing vector processors can achieve speeds nearly 100 times those of the conventional scalar processors found in most central computing facilities. Furthermore, current studies are being made of processors for the 1980's that have 1000 times present conventional speeds. The need for such high-speed computation is quite clear in certain scientific and engineering applications, especially involving 3-dimensional and transient phenomena in physical systems with coupled thermodynamic, fluid, mechanical, and/or electrical effects.

This paper is intended to introduce the simulation enthusiast in a discipline not included in the above to the concepts and potential advantage of vector processing.

Before proceeding with the technical discussion, it is worth considering a common characteristic of most large scale simulation problems.

To justify the need for massive computation, simulation programs in general require massive amounts of data on which to perform the computation (the complexity of most computation being limited to O(n) where n is the number of data elements). One source of such data is automated measurement devices; another is the algorithmic generation of large data sets from small ones as occurs in the production of large matrices for solution of partial differential equations, given only a small set of physical dimensions and constants. Data sets - and hence computation - dependent on personal collection or other non-automated production will inherently be limited in size, due principally to the time for collection and entry of the data into the machine.

This algorithmic generation of large data sets nearly always implies a regular problem structure. Thus, large problems inherently contain the solution structure to be required for vector processing.

DEFINITIONS

The rather special viewpoint of this paper is indicated by the following definitions.

Scalar processor - a machine that processes single and array data elements with similar speed.

Vector processor - a machine that processes array data elements at a higher rate than single elements.

It should be noted that (a) the computer architecture to achieve this speedup is of no consequence (b) the appearance of array constructs in a language (such as APL) is not related to the speedup issue and (c) the availability of high level programming languages that exploit this speed is assumed.

Such vector processing characteristics would be indicated in Fortran (for example) by the ability to process preferentially either

single loop

```
      DO 1 J=1,N
    1 A(J)=B(J)+C(J)
```

multiple loop

```
      DO 1 J=1,N
      DO 1 K=1,N
    1 A(J,K)=A(J,K)+B(J)*C(K)        (1)
```

in one vector operation.

One method of achieving this speedup is through the use of parallel processors Figure (1). It is easy to visualize that the single loop above could be executed in one parallel step by N processors.

MATHEMATICAL DESCRIPTION

Although not a universal mathematical description of all vector processors, the following model provides a basis for deciding whether an algorithm is amenable to vector processing.

In general, the time to perform a single- or multi-loop operation is given by the formula

$$T_v = T_s + \ell T_{op} \qquad (2)$$

where T_s is the time to startup the vector operation, T_{op} is the time to complete an arithmetic (or logical) computation, and ℓ is the vector length (N in Equation (1)). The corresponding vector timing diagram is given in Figure (2). The comparison with a seemingly slower scalar processor is also shown. Note that the vector startup is shown to make vector processing potentially slower for short vectors; the crossover point A occurs typically for array lengths between 2 and 5.

ALGORITHM EVALUATION

To decide whether a particular algorithm is amenable to vectorized solution, the following compact measure is introduced.

Assume that the algorithm involves only m vector operations, the ith operation having length ℓ_i, startup time T_s, and arithmetic operation time T_{op}. Since time T_{op} is the useful computation time, define the vectorization efficency as

$$m = \frac{\text{arithmetic time}}{\text{startup time + arithmetic time}} = \frac{1}{1 + \frac{1}{L_{ave}}(T_s/T_{op})}$$

where

$$L_{ave} = \sum_{i=1}^{m} \ell_i/m$$

is the average vector length. Note that
(a) T_s/T_{op} is a processor parameter (typically between 10 and 100) whereas L_{ave} is a characteristic of the algorithm only;
(b) a large L_{ave} yields a higher efficiency;
(c) when $L_{ave} = T_s/T_{op}$, one *half of the* computation time is devoted to useful (arithmetic) computation.

The latter property allows a compact representation of the vectorizeability of competing algorithms or of a single algorithm applied to a family of problems. For example, in (3) a computationally efficient scheme for solving families of finite element problems arising from partial differential equation solutions is evaluated for its vector characteristics. In Figure (3), the average vector length is plotted versus grid size for several families of finite elements. It is immediately clear that, whereas case A has marginal vector characteristics for $10 \le T_s/T_{op} \le 100$, case B is an excellent candidate for solution on any vector processor.

PROCESSOR/ALGORITHM EVALUATION

Once an algorithm is judged amenable to vector processing, a means of evaluating its performance on specific vector processors is required. The idealizations inherent in the characterization of an algorithm by L_{ave} are now removed. For example, a study of the timings of vectorized algorithms shows that appreciable time may be spent on unavoidable scalar operations which preamble the vector portions of the code. Also, conflicts may arise in the routing of data on a particular machine.

Despite this deviation from idealized performance, it is generally true that larger problems - involving large array operations - exploit the algorithm and processor vector attributes most fully. In the limit, as problem size grows to infinity, the combined performance is usually governed only by (a) the number of arithmetic operations N_{ar}, and (b) the operation time T_{op}. Thus, if n is the problem size and T is the total computation time, the operation <u>rate</u> R

$$R(n) = \frac{N_{ar}(n)}{T(n)}$$

has the property that

$$R(\infty) = \lim_{n \to \infty} \frac{N_{ar}(n)}{T_{op} N_{ar}(n)} = 1/T_{op}$$

If R(n) were to be plotted versus n, this representation would have two properties:

(a) an asymptotic value of $1/T_{op}$, and
(b) an initial value $R(0) = 0$, due to program overhead, and
(c) a usually monotonic increase to $1/T_{op}$

The rate at which R(n) approaches its asymptotic value gives an indication of the performance of the algorithm/architecture on small problems. An ideal characteristic would be $R(1) = 1/T_{op}$. The next most favorable shape would be a scalar processor characteristic, where the startup time T_s in Equation (2) would be zero.

Comparisons have been made of several of the current commercial vector processors, based on their floating point processing rate in solving systems of simultaneous linear equations, using two different classes of algorithms (Figure (4)). Although such benchmarks are subject to qualification due to the different numeric precision and different programming languages involved (1), this display clearly shows certain processors with improved small-problem performance. This observation has since been supported by tests on a number of algorithms in reference (2).

IMPACT ON USER

A review of current vector processors reveals a distinction between "Fortran" and "assembly language" processors. At present, a design tradeoff can be made between a machine with a variety of vector-related hardware resources that must be controlled from an assembly or non-Fortran language, and machines which can respond only to standard Fortran constructs. Although clearly the "Fortran" machine cannot be faster than the former, it is unclear at this time what the tradeoff will be in attempts to achieve the very high processing rates of the 1980's.

IMPACT ON MODELING

Perhaps more in keeping with this conference is the impact of machine architecture on modeling. Although a regularity in the problem structure is necessary, this regularity may take several forms.

(1) In Figure (5a), a "sparse" system is illustrated. Here, a large number of small systems are loosely interconnected. In this case, each vector (array) would contain an element from each system, so that an array operation would contribute to the partial solution of every small system.

(2) In contrast, the two systems of Figure (5b) would be solved in sequence, on the assumption that each system is sufficiently dense to generate long vectors in its solution (e.g., row operations used in solution of a large matrix)

A more subtle distinction between these two model systems occurs, however, when one examines in detail the data flow necessary to carry out their solutions. In the sparse system of Figure (5a), the small system solutions must be combined according to the interconnection pattern. Since the systems are themselves quite small, the ratio of data movement (corresponding to the interconnection) to computation (system solution) is less than for the dense system. The routing of data in a vector architecture of the form of Figure (1) can be difficult under any circumstances, and could easily dominate the arithmetic computation on current vector processors for such as "sparse" systems.

CONCLUSION

The impact of vector processors on the general simulation field cannot be evaluated at this time. Although one is tempted to note that, like hybrid computers, vector machines will initially be available to only a few research groups and will require rather specialized programming efforts, the analogy is difficult to carry further for several reasons.

First, hybrid computer manufactures are themselves examining the possibility of replacing hybrid configurations with small vector processors architected to solve common simulation problems efficiently. In such an event, the class of problems now analyzed with hybrid computers would become a subset of problems solveable on the new vector processors.

Second, it is reasonable to assume that computer architects will strive to reduce the impact of vector-related parameters such as startup time on computational performance and that system software developers will reduce the problem of user control of architectural features.

REFERENCES

(1) Calahan, D. A., Joy, W. N., and Orbits, D. A., "Preliminary Report of Matrix Benchmarks on Vector Processors," Report #94, Systems Engineering Laboratory, University of Michigan, May, 1976.

(2) Keller, T. W., "Cray-1 Evaluation; Final Report," Report LA-6456-MS, Los Alamos Scientific Laboratories, December, 1976.

(3) Calahan, D. A., "Complexity of Vectorized Solution of 2-Dimensional Finite Element Grids," Report #91, Systems Engineering Laboratory, University of Michigan, November, 1975.

(4) Calahan, D. A., and Orbits, D. A., "Data Flow Considerations in Implementing a Full Matrix Solver with Backing Store on the Cray-1," Report #98, Systems Engineering Laboratory, University of Michigan, September, 1976.

ACKNOWLEDGMENT

This work was sponsored by the Directorate of Mathematical and Information Sciences, Air Force Office of Scientific Research under Grant 75-2812.

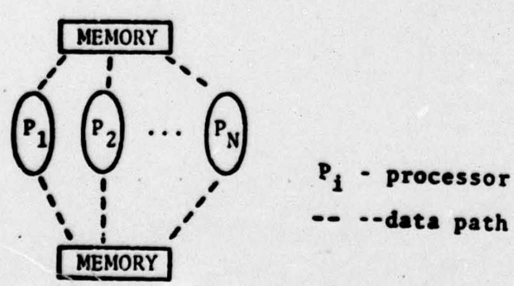


Figure 1. Parallel processor vector architecture

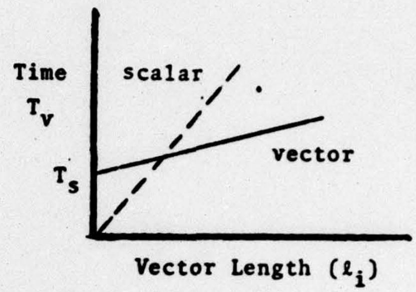


Figure 2. Vector timing diagram

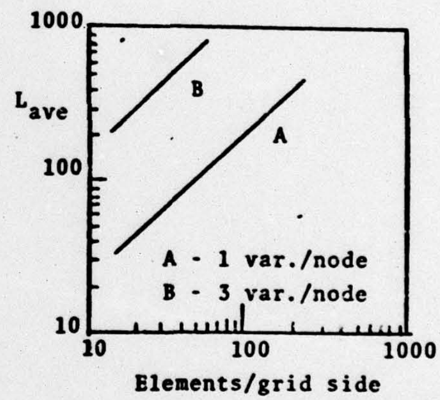


Figure 3. Average vector length for linear finite element solution

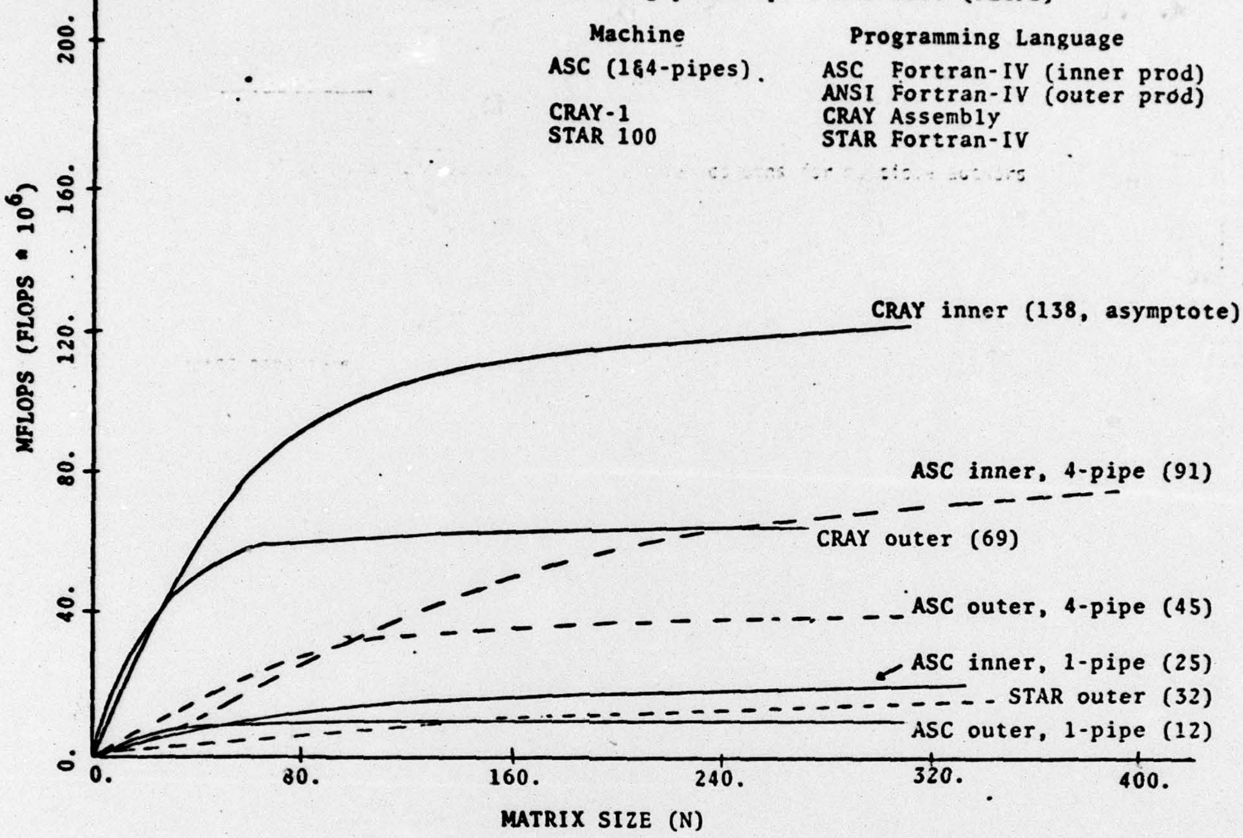
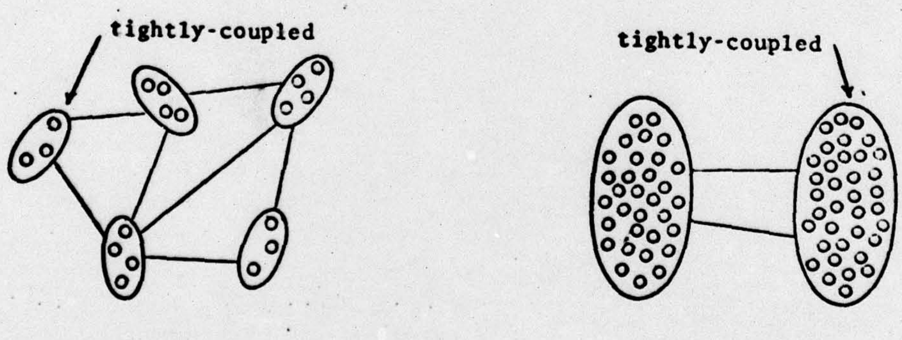


Figure 4. Floating point operation rate (FLOPS)



(a)Many small systems

(b)Few dense systems

Figure 5. Illustration of sparse,dense systems